C PROGRAMLAMA DİLİNE GİRİŞ

C Dilinin Tarihçesi

- 1972'de 'Dennis Ritchie' tarafından AT&T Bell Laboratuarlarında B dilinden geliştirildi.
- C dili konusundaki ilk kitap 1978'de 'Dennis Ritchie ve Brain Kernighan' tarafından yazıldı.
- Bu tarihten sonra bir çok C uygulaması geliştirildi.
- Belli bir standarda göre hareket etmediklerinden uygulamalar arasında farklılıklar ortaya çıktı.
- 1983 de ANSI (American National Standard Institute) C standardı oluşturuldu.

C Dilinin Özellikleri

- C dili donanımdan bağımsız (machine independent) ve taşınabilirdir (portable).
- Taşınabilirlik, aynı programın farklı donanımlarda ve farklı işletim sistemlerinde işletilebilirliğini ifade eder.
- Bir dilin verimliliği (productivity), hızlı fakat fazla yer kaplamayan yazılımlar geliştirmeye olanak sağlaması ile ölçülür.
- C Programlama dili verimli, basit, fakat güçlü yapısından ve bu dilde geliştirilen uygulamaların taşınabilir olmasından dolayı yaygın olarak kullanılmaktadır.

C Programlarının Temel Yapısı

- Bir C programı iki temel bölümden oluşur;
  - Önişlmeci Direktifleri (Preprocessor Directives) .
  - Ana Fonksiyon (main function)

Önişlemci Direktifleri

- Önişlemci direktifleri # işareti ile başlar ve program derlenmeden önce C önişlemcisi tarafından işletilir.
- Her bir önişlemci direktifinin farklı bir görevi vardır.
- **#include** ve **#define** en çok kullanılan direktiflerdir.
- **#include** direktifi program içerisinde kullanılan fonksiyonlar için gerekli kodları programa dahil etmek için kullanılır.

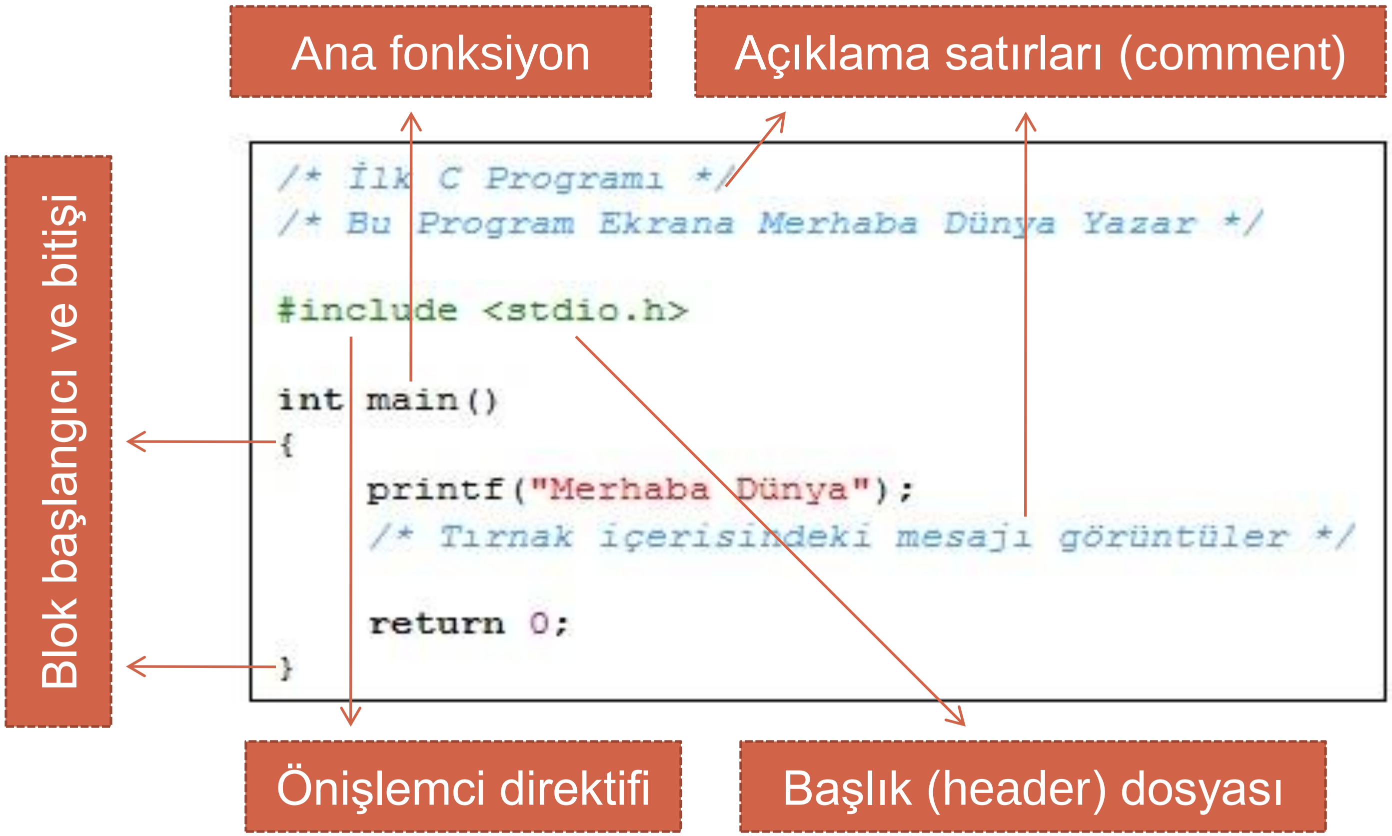
Önişlemci Direktifleri

- Örneğin tasarladığımız programda ekrana çıktı yazdırmak istiyoruz.
- Bunun için C dilinin standart bir fonksiyonu olan **printf** fonksiyonunu kullanmamız gerekir.
  - **Printf("Örnek Çıktı");**
- Ancak **printf** fonksiyonunun çalışabilmesi için **<stdio.h>** isimli dosyaya ihtiyacımız olacaktır.
- Bu dosyayı programa dahil etmek için program kodunun en tepesine
  - **#include <stdio.h>** komut satırı yazılır.
- C dilinde **.h** uzantılı dosyalara başlık dosyası (**Header File**) adı verilir.
- **stdio.h** başlık dosyası standart giriş çıkış işlemleri için gerekli kodları içerir.

Main Fonksiyonu

- Hemen hemen bütün C programları birden fazla fonksiyondan oluşur.
- **main()** bütün C programlarında bulunması gereken programın ana fonksiyonu yani gövdesidir.
- İlk çalıştırılacak olan fonksiyondur.
- Programda çalıştırılacak ifadeler (kod satırları) **{ - }** küme parantezleri içinde yazılırlar.
- Her parantez çiftinin oluşturduğu yapılara kod blokları denir.
- Bir kod bloğu içerisinde program içerisinde kullanılacak değişkenler ve gerçekleştirilecek işlemleri yerine getirecek komutlar bulunur.

Basit Bir C Programı



Ekrana “Merhaba Dünya” yazan bir program

C dilinin genel yazım kuralları

- C dili birden fazla satırdan oluşan açıklama satırlarına izin verir.
- Bu açıklama satırları programın ne iş yaptığı hakkında bilgi verir.
- Açıklama satırları **/*** ile başlayıp ***/** ile sona erer.
- Derleyici bu satırları çalışma anında dikkate almaz.
- C de her bir işletilebilir ifade (komut satırı) **;** ile sonlandırılır.
- Bütün anahtar kelimeler ve komutlar küçük harfle yazılır (**#define** hariç).
- C dili büyük-küçük harf duyarlıdır.
  - Yani; "TOPLAM", "toplam" ve "tOpLaM" kelimelerinin hepsi C derleyicisi tarafından ayrı ayrı algılanır.

İşleçler (Operator)

- İşleç (**operatör**), matematiksel, mantıksal ve atama işlemlerini gerçekleştirmek için kullanılan simgelerdir.
- Bu işleçler ile işleme giren değerlere işlenen (**operand**) adı verilir.
- **Operatörler** birden fazla **operand** alabilirler.
- Tek **operand** alanlara **unary operatörler** denilir.
- C dilinde kullanılan **operatörler** üç grupta incelenir.
  - Matematiksel, Mantıksal ve Atama operatörleri.

Matematiksel operatörler

Matematikteki ifade	C dilindeki ifade	işlem
.X veya hiçbir şey	*	çarpma
Mod	%	modüler bölme
/	/	bölme
+1	++	1 artırma (unary increment)
-1	--	1 eksiltme (unary decrement)
+	+	Toplama
-	-	çıkarma

Unary operatörlerin ifadenin sağında yada solunda olması ile anlam değişik olur.	
A++ (postincrement)	Önce A'yı kullan sonra içindeki değeri 1 artır.
++A (preincrement)	Önce A'yı 1 artır sonra bu artırılmış değeri kullan.
A-- (postdecrement)	Önce A'yı kullan sonra içindeki değeri 1 azalt.
--A (predecrement)	Önce A'yı 1 eksilt sonra bu eksiltilmiş değeri kullan.

Matematiksel operatörler

Unary operatörlerin ifadenin sağında yada solunda olması durumunda oluşan anlam farklılığına bir örnek;

a=4 , b=5 olsun.

İşlem	İşlemlerin gerçekleştirilme sırası		Değişkenlerin son değerleri		
c=a++ *b	c=a*b a=a+1	c=4*5=20 a=4+1=5	a=5	b=5	c=20
c=++a –b	a=a+1 c=a-b	a=4+1=5 c=5-5=0	a=5	b=5	c=0
c=--a + b--	a=a+1 c=a+b b=b-1	a=4-1=3 c=3+5=8 b=5-1=4	a=3	b=4	c=8

Mantıksal operatörler

Matematikteki ifade	**C dilindeki ifade**	**işlem**
>	>	Büyük
<	<	Küçük
≤	<=	Küçük veya eşit
≥	>=	Büyük veya eşit
=	==	Eşit (karşılaştırma)
≠	!=	Eşit değil
^	&&	Ve (And)
V	||	Veya (Or)
‘	!	Değil (not-invert)

Mantıksal operatörler

Ve (&&)

A	B	işlem	sonuç
Hayır	Hayır	A && B	Hayır
Hayır	Evet	A && B	Hayır
Evet	Hayır	A && B	Hayır
Evet	Evet	A && B	Evet

Veya (||)

A	B	işlem	sonuç
Hayır	Hayır	A || B	Hayır
Hayır	Evet	A || B	Evet
Evet	Hayır	A || B	Evet
Evet	Evet	A || B	Evet

Atama operatörleri

Operatör	İşlem
=	Atama
+=	Toplayarak atama
-=	Eksilterek atama
*=	Çarparak atama
/=	Bölerek atama
%=	Mod alarak atama

Değişken tanımlamaları (Definition) ve Veri türleri (Data Type)

- Programda kullanılacak bütün değişkenlerin kullanılmadan önce C derleyicisine bildirilmesi gerekir.
- Değişkenleri derleyiciye bildirme işine değişken tanımlama (variable definition) denir.
- Değişken tanımlamak, değişken için bir isim seçmek ve içerisinde ne tür bir veri saklanacağını söylemekle yapılır.
- Her değişkenin ismi, veri türü ve değeri vardır.
- Değişken ismi değişken saklayacak bellek hücresine verilecek isim.
- Veri türü saklanan bilginin türü (ondalık sayı, tam sayı..).
- Değeri ise değişken içinde saklanan bilginin kendisidir.

Değişken isimlendirme

- Programda kullanılacak değişken isimleri programcı tarafından tanımlanır.
- Değişkenlere isim verirken aşağıdaki kurallara uyulmalıdır.
  - C dilinin kendine özgü anahtar sözcükleri, komut veya fonksiyon adları değişken ismi olarak kullanılamaz.
  - Değişken isimleri içerisinde, a-z ve A-Z arası İngilizce harfleri, 0-9 arası rakamlar ve özel karakter olarak sadece alt çizgi (_) karakteri kullanılabilir.
  - Özel karakterler (+,-,! vs.) ve Türkçe karakterler kullanılmaz.
    - Maaş, öğrenci, sınıf -> bunlar değişken adı olarak kullanılamaz.
  - Değişken ismi rakam ile başlayamaz.
    - 1. vize (yanlış) vize1 (doğru)

C dilinin bazı Anahtar Sözcükleri

ANSI C (C89)/ISO C (C90) keywords:

- `auto`
- `break`
- `case`
- `char`
- `const`
- `continue`
- `default`
- `do`
- `double`
- `else`
- `enum`
- `extern`
- `float`
- `for`
- `goto`
- `if`
- `int`
- `long`
- `register`
- `return`
- `short`
- `signed`
- `sizeof`
- `static`
- `struct`
- `switch`
- `typedef`
- `union`
- `unsigned`
- `void`
- `volatile`
- `while`

Veri türleri

- Veri türü, değişken içerisinde tutulan değerin tipini belirler (tamsayı, ondalık sayı, karakter, vs.).

- C dilinde dört temel veri tipi (türü) vardır.
  - **char**, **int** , **float** ve **double**.

- **char:** bir harf, bir rakam veya özel bir simgeyi saklamak için kullanılan veri tipidir.
  - **char** tipindeki her bir değer tek tırnak arasına yazılmalıdır (‘A’, ‘z’, ‘2’, ‘*’, ‘:’, ‘ ’).
  - **char** tipinde tanımlanan bir değişkenin tuttuğu değer karakterin ASCII tablosundaki sayısal karşılığıdır.

- **int:** -32768 ile +32768 (Dos işletim sisteminde) arasındaki tam sayı değerlerini saklamak için kullanılır.

- **float:** ondalıklı olarak verilen sayıları ifade edebilmek için kullanılır.

- **double:** ondalıklı olarak verilen çok küçük ve çok büyü sayıları ifade edebilmek için kullanılır. float türünden iki kat daha fazla duyarlılığa sahiptir.

Veri tipleri, kapladığı alanlar (size) ve değer aralıkları (range)

ÇIKTI

Windows (32 bit) Turbo C	Windows (32 bit) Salford	Linux (32 bit) GCC	Linux (64 bit) GCC
char : 1 bayt	char : 1 bayt	char : 1 bayt	char : 1 bayt
short : 2 bayt	short : 2 bayt	short : 2 bayt	short : 2 bayt
int : 2 bayt	int : 4 bayt	int : 4 bayt	int : 4 bayt
long : 4 bayt	long : 4 bayt	long : 4 bayt	long : 8 bayt
unsigned char : 1 bayt	unsigned char : 1 bayt	unsigned char : 1 bayt	unsigned char : 1 bayt
unsigned short : 2 bayt	unsigned short : 2 bayt	unsigned short : 2 bayt	unsigned short : 2 bayt
unsigned int : 2 bayt	unsigned int : 4 bayt	unsigned int : 4 bayt	unsigned int : 4 bayt
unsigned long : 4 bayt	unsigned long : 4 bayt	unsigned long : 4 bayt	unsigned long : 8 bayt
float : 4 bayt	float : 4 bayt	float : 4 bayt	float : 4 bayt
double : 8 bayt	double : 8 bayt	double : 8 bayt	double : 8 bayt
long double : 10 bayt	long double : 10 bayt	long double : 12 bayt	long double : 16 bayt

Veri Tipi	Açıklama	Bellekte işgal ettiği boyut (bayt)	Alt sınır	Üst sınır
char	Tek bir karakter veya küçük tamsayı için	1	-128	127
unsigned char			0	255
short int	Kısa tamsayı için	2	-32,768	32,767
unsigned short int			0	65,535
int	Tamsayı için	4	-2,147,483,648	2,147,483,647
unsigned int			0	4,294,967,295
long int	Uzun tamsayı için	8	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
unsigned long int			0	18,446,744,073,709,551,615
float	Tek duyarlı gerçel sayı için (7 basamak)	4	-3.4e +/- 38	+3.4e +/- 38
double	Çift duyarlı gerçel sayı için (15 basamak)	8	-1.7e +/- 308	+1.7e +/- 308

Değişken tanımlama yerleri, şekilleri ve ilk değer atama

- Standart C de üç farklı yerde değişken tanımlanabilir.
- fonksiyonların üstünde, blokların { } içerisinde ilk sırada ve fonksiyonlarda parametre olarak.
- İlk değer ataması yapılmayan değişkenlerin değerleri (eğer main fonksiyonunun üstünde tanımlanmışsa) sayısal olanlar 0 diğerleri boş olarak belirlenir, eğer main içinde tanımlanmışsa bellekte rastgele değerler olarak belirlenir.

```
#include<stdio.h>

/* Örnek Değişken tanımlamaları */

int main(){

    int a;        →  int a=1;  →  İlk değer atama
    a=1;

    int x, y, z; /* Aynı satırda birden fazla değişken tanımlanabilir. */

    char m='k';

    double n=5.05;

}
```

Genel giriş-çıkış fonksiyonları

- Geliştirilen bir programda kullanıcıdan bilgi (veri) almak ve kullanıcıya bilgi göstermek en temel kullanıcı etkileşim işlemleridir.

- C dilinde bütün giriş-çıkış operasyonları giriş ve çıkış fonksiyonları (input-output functions) ile gerçekleştirilir.

  - C de kullanılan standart giriş-çıkış fonksiyonları, **printf()** ve **scanf()** dir.

- Bu fonksiyonlar C standart giriş çıkış kütüphanesinde **stdio.h** bulunurlar.

- Giriş-çıkış fonksiyonlarını program içerisinde kullanabilmek için **stdio.h** başlık dosyasının programa dahil edilmesi gerekir.

  - #include <stdio.h>

printf() fonksiyonu

- Değişkenlerin değerlerini, hesaplanan sonuçları yada mesajlar ekranda göstermek için kullanılır.
- printf() fonksiyonu, fonksiyon ismi ve parantezler içindeki parametreler olmak üzere iki kısımdan oluşur.
- printf() fonksiyonu, parametre olarak görüntülenecek bilginin hangi biçimde görüntüleneceğini bildiren çıktı metin formatını ve bu formatın içinde yazdırılacak olan değişkenler listesini alır.
- Genel kullanım formatı aşağıdaki gibidir.
  - printf(çıktı metin formatı, değişken listesi)
    - printf(“Hesaplanan Yaşınız: %d dir \n”, yas);
  - printf(çıktı metin formatı)
    - printf(“Merhaba Dünya”);

printf() fonksiyonu

- printf çıktı formatındaki % karakterinin dışındaki tüm karakterleri ekrana yazar.
- % karakterini gördüğünde bunun sağındaki karakteri yazdırılacak değişkenin format karakteri olarak ele alır.
- printf çıktı formatındaki \ karakterine escape karakteri denir.
- Bu karakterden sonra gelen karakter ise escape serisini ifade eder.
  - Örneğin \n ifadesi, çıktı ekranında yeni bir satıra geçilmesi gerektiğini ifade eder.

Escape serileri	Anlamı
\n	Yeni satıra geç.
\t	Bir sonraki sekme durağına (tab stop) geç.
\a	İkaz (alert). Beep sesi
\\	Ters bölü (black slash) işaretini yazdırmak.
\”	Tırnak işaretini yazdırmak içi.

scanf() fonksiyonu

- Değişkenlerin içerisine klavyeden değer atamak için kullanılır.
- Fonksiyon ismi ve parametrelerden oluşur.
- Parametre olarak, girilecek değerin hangi formatta olacağını bildiren girdi formatını ve bu formata göre girilecek değişkenler listesini alır.
- scanf fonksiyonunda dışarıdan değer girilecek bütün değişkenlerin başına & işareti konur.
- Bu işaret bellek operatörüdür, değişkenlerin tutulduğu bellek hücresinin adresini okur.

Format karakterleri

Format Karakterleri	Anlamı
d	Tamsayılar için (onluk sistemde)
i	Tamsayılar için (onluk, sekizlik ve onaltılık sistemde)
lx	Uzun tamsayıları onaltılık (hex) sistemde
ld	Uzun tamsayılar için (onluk sistemde)
o	Tamsayılar için (sekizlik sistemde)
x	Tamsayılar için (onaltılık (hex) sistemde)
X	Tamsayılar için (onaltılık (hex) sistemde büyük harfle)
u	İşaretsiz tamsayılar için
c	Char tipinde değişkenler (karakterler) için
s	Metin (strin)
f	Float değişkenler için
E , e	Double ve float türlerini üstel gösterimde yazmak için
lf	Double değişkenler için

printf basit bir uygulama

```c
/*
  printf() fonksiyonunun çalışmasına örnek.
*/

#include <stdio.h>

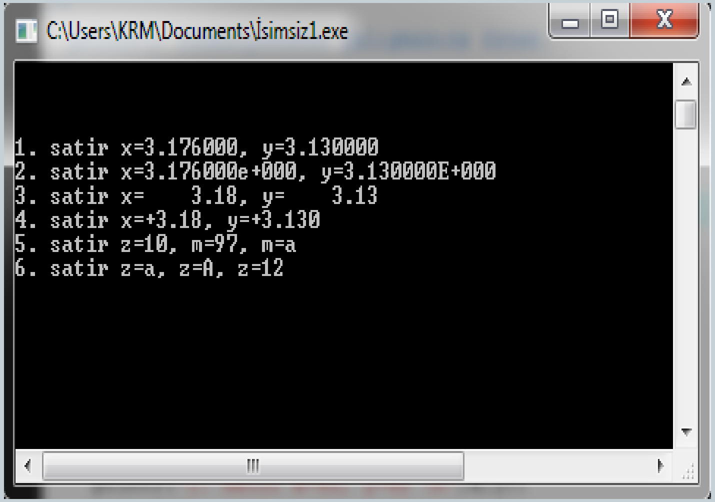
int main(){

    double x=3.176;
    float y=3.13;
    int z=10;
    char m='a';

    printf("\n\n\n");
    printf("1. satır x=%f, y=%f \n",x,y);
    printf("2. satır x=%e, y=%E \n",x,y);
    printf("3. satır x=%8.2f, y=%8.2f \n",x,y);
    printf("4. satır x=%+.2f, y=%+0.3f \n",x,y);
    printf("5. satır z=%d, m=%d, m=%c \n",z,m,m);
    printf("6. satır z=%x, z=%X, z=%o \n",z,z,z);

    getchar();

    return 0;
}
```



scanf basit örnek

```c
/*
  Taban ve Yükseklik değerine göre
  *alan hesabı yapan program
*/

#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{

  int taban;
  int yukseklik;
  float alan;

  printf("Ucgenin Tabani = ");
  scanf("%d",&taban);

  printf("Ucgenin Yuksekligi = ");
  scanf("%d",&yukseklik);

  alan = ( taban  * yukseklik ) / 2.0;

  printf("\n Ucegign Alani %.2f dir \n", alan);

  system("PAUSE");
  return 0;
}
```

